Bir web sayfasında header,body ve footer olmak üzere 3 ana yapı bulunmaktadır.Header kısmı genelikle menü çubuğu;footer kısmında sayfayla ilgili bilgi kısımları,iletişim ve harita gibi bilgiler bulunur ve header,footer kısımları sayfada genellikle sabittir.Bununla beraber body kısmı ise dinamik ve değişen ksıımdır.Sayfada menü kısımlarına bastığımızda ordaki bilgileri bize getirir mesela.

Oluşturduğumuz temel mvc yapısında view klasörü içeerisinde oluşturulan shared klasöründe sayfanın ana yapısı bulunmakta ve tüm sayfa aslında temel olarak bu .cshtml uzantılı sayfadan yönetilerek katmanlara ayrılmaktadır ve controller yapıları sayesinde ActionResulttlar ile View’ler oluşturulabilir ve tüm bu view’ler \_Layout.cshtml sayfasından yönetilir ve ona bağımlıdır.Controller ekleyerek her bir menü durum çubuğu için Action Result tanımlanır ve çeşitli view’ler oluşturulur. \_Layout.cshtml sayfasında RenderBody,footer ve header kısımlarının ana özellikleri bulunur. @RenderBody değişken alandır.cshtml Renderbody ile alınıp sayfaya basılır.Web sitenin başlama sayfası appStart klasör altındaki RouteConfig kısmında yazan adresle başlar.

MVC yapısındaki parçaların görevleri temel olarak şu şekildedir :

* **Model** – Uygulamada kullanılacak verilerin bulunduğu, veritabanı ile ilgili bağlantının yapıldığı katmandır.
* **View** – Model içerisindeki verilerin görselleştirilmesinden sorumlu katmandır.
* **Controller** – Model ile View arasındaki bağlantıyı sağlayan katmandır. View’dan gelen ekleme, silme, güncelleme vb. isteklere cevap verir

## **ASP.NET MVC nedir?**

Web teknolojilerinin gelişmesiyle birlikte 2007 yılında ASP.NET içerisine eklenen bir yazılım mimarisidir.Geliştirilen web uygulamalarındaki bölümlerin artmasıyla beraber yazılan kodlarda artmaktadır.Yazılan kodların artması, geliştirilen uygulama için uygun bir mimarinin seçilmemesi kodların karmaşıklaşmasına, kod bakım süresinin uzamasına neden olur.Microsoft bu sorunu çözmek için web uygulamalarında sıklıkla kullanılan Mvc mimarisini Asp.Net içerisine entegre etmiştir.Bu yapı temel olarak uygulamaları parçalara ayırır.Parçalara ayrılmasındaki neden seperation of conserns yani sorumlulukların ayrılması prensibidir.

Örnek olarak,kullanıcı View katmanındaki görsel form alanı aracılığıyla ekleme işlemi için veri gönderir ve veriler Controller katmanında kontrol edilir.Kontrol edilen veriler uygunsa Model katmanına aktarılarak ekleme tamamlanır.Kontrol başarısızsa View katmanına geri bildirim gider.

Controller katmanındaki kontrol işlemleri zorunlu değildir.

Bu klasörde MVC temel iskeleti oluşturulmuş ve kaynak kodlar verilmiştir.Çıkan görsel sonuç yani web sayfasının iskeleri ve 3 ana alanımız ise aşağıdaki resimde gösterilmiştir.

metin, ekran görüntüsü, yazılım içeren bir resim

Açıklama otomatik olarak oluşturuldu

MVC TEMEL İSKELET